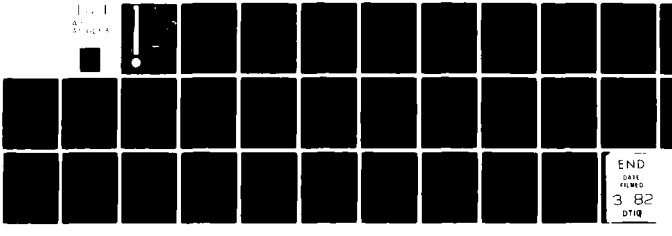
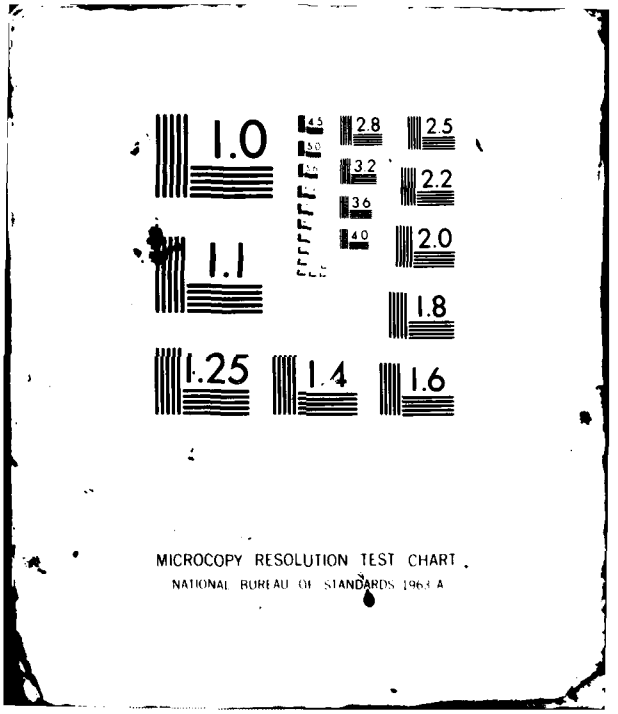
AD-A110 253 STATE UNIV OF NEW YORK AT BUFFALO AMHERST DEPT OF CO--ETC F/G 12/1
MORPH-FITTING, AN EFFECTIVE TECHNIQUE OF APPROXIMATION,(U)
NOV 81 N V FINDLER, E MORGADO AFOSR-81-0220

UNCLASSIFIED AFOSR-TR-81-0893 NL



END
DATE FILMED
3 82
DTIC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

LEVEL II

(3)

MORPH-FITTING -- AN EFFECTIVE TECHNIQUE OF APPROXIMATION*

BY

NICHOLAS V. FINDLER
AND ERNESTO MORGADO

NOVEMBER 1981

DEPARTMENTAL TECHNICAL REPORT NUMBER: # 188

GROUP FOR COMPUTER STUDIES OF STRATEGIES
TECHNICAL REPORT NUMBER: # 2

*THE WORK DESCRIBED HAS BEEN SUPPORTED BY THE AIR FORCE OFFICE OF SCIENTIFIC RESEARCH [redacted] AFOSR-81-0220

TO APPEAR IN IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, SPECIAL ISSUE: COMPUTER INTELLIGENCE -- THREE DECADES; MARCH 1982

STATE UNIVERSITY OF NEW YORK AT BUFFALO

Department of Computer Science

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFOSR-TR- 81-0893	GOVT ACCESSION NO.: AD-A120 153	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): MORPH-FITTING -- AN EFFECTIVE TECHNIQUE OF APPROXIMATION		5. TYPE OF REPORT & PERIOD COVERED: INTERIM, 1 JUL 80 - 30 JUN 81
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Nicholas V. Findler and Ernesto Morgado		8. CONTRACT OR GRANT NUMBER(s): AFOSR-81-0220
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Department of Computer Science, State University of New York at Buffalo, 4226 Ridge Lea Road, Amherst NY 14226		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F; 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS: Directorate of Mathematical & Information Sciences, Air Force Office of Scientific Research, Bolling AFB DC 20332		12. REPORT DATE: NOV 81
		13. NUMBER OF PAGES: 22
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report): UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Morphs, basic patterns, Morph-fitting, numerical approximation, sequential regression analysis, noise-tolerant and noise-controlled approximation technique.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An algorithm has been developed and programmed which fits a minimum number of basic patterns, morphs, to a sequence of datapoints. The dependent variable is given as a scalar value; the independent variable is 'distance-like' or 'time-like' and can represent continuous or discrete values or an event-counter. The morphs fitted are an indefinite number of occurrences of trends (straight lines) step functions, and sudden changes (peaks of short duration).
(CONTINUED)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 412759

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ITEM #20, CONTINUED:

Delay functions span over periods characterized by uninterpretable events.

A useful by-product of the investigation is a set of optimum decision rules concerning the boundary points between sequential regression functions.

MORPH-FITTING -- AN EFFECTIVE TECHNIQUE OF APPROXIMATION

Nicholas V. Findler and Ernesto Morgado*

Group for Computer Studies of Strategies

Department of Computer Science

State University of New York at Buffalo

4226 Ridge Lea Road

Amherst, NY 14226

ABSTRACT

An algorithm has been developed and programmed which fits a minimum number of basic patterns, morphs, to a sequence of data points. The dependent variable is given as a scalar value; the independent variable is "time-like" and can represent continuous or discrete time or an event counter. The morphs fitted are an indefinite number of occurrences of trends (straight lines), step functions, and sudden changes (peaks of short time duration). Delay functions span over periods characterized by uninterpretable events.

A useful by-product of our investigations is a set of optimum decision rules concerning the boundary points between sequential regression functions.

*The development of the algorithm was joint effort. The program based on it was produced by E.M. and the paper was written by N.V.F.

1. THE TASK OF THE ALGORITHM IN GENERAL

Regression, Fourier and time-series analyses are the three most well-known and widely-used techniques to describe (or -- as often referred to -- explain, interpret and predict) the behavior of some observable/measurable variables, the "dependent" variables, in terms of some others, the "independent" variables, which are normally assumed to be known exactly. In spite of the powerful mathematical machinery developed for these methods, there are certain types of tasks that are not easily handled by even the combination or extension of such methods (cf. spline functions). A relatively simple and special case is the problem, so far not solved in a satisfactory and easily applicable manner, of where the boundary points should be between subsequent segmented (or multiphase) regression lines (see, e.g., [1-5]).

In our research on decision making under uncertainty and risk (see, e.g., [6-14]), we have studied a variety of techniques for the analysis and synthesis of competitive strategies. One of these investigations has been aimed at a system which can identify rules of pattern formation, recognize stochastic relations between patterns of causally associated variables, and use such relations to estimate the values of variables that are hidden from observation most of the time [15, 16].

Programs of this type, capable of making inductive inferences over large domains of phenomena, would be of great value in interpreting experimental data and in detecting patterns which underlie the rules governing a given body of observations. The first phase of this project has been the creation of a

Pattern Identifier and Categorizer (PIC) program. It describes in mathematical terms the sequence of values of any open variable selected so that certain characteristic features of the descriptions can then be made to relate, presumedly in a causal manner, to the hidden variables of the environment. This is now briefly discussed next.

The history of an environment can be characterized by an event vector $\vec{E}$, each component of which is a function of a "time-like" variable. The latter can be discrete (quantized) or continuous time, or an event counter. (For the sake of simplicity, we shall refer to continuous time in the present explanation.) The values of each component, e_i, are scalar and are measured at arbitrary (not necessarily equally-spaced) time points.

In order to interpret, or mathematically "explain", the history of the environment, $\vec{E}[e_1(t), e_2(t), \ldots]$, PIC has to generate a uniform structure which can reproduce each e (t) within tolerable limits. From among the three mathematical techniques mentioned above, the time-series analysis may be the most appropriate for such tasks. However, we have decided to develop a more flexible and computationally less expensive method. The idea is to select a small set of basic patterns, called by us morphs, and use an indefinite but minimum number of occurrences of these to fit $e_i(t)$ so that the "unexplained variance" (the sum of the squares of the difference between the values of e_i, and the morph) is also minimum.

Figure 1 shows the set of morphs, each with its

characteristic parameters:

- -

FIGURE 1 ABOUT HERE

- -

A trend is a straight line (monotonic increase or decrease) over a certain range of t. The three parameters, a, b and c, completely define it. (The y-coordinates of the starting point of a trend and of the end point of the previous morph are identical. The first trend is always preceded by a step function.)

A step function, determined by its two parameters a and b, is an instantaneous increase in the event vector component.

A sudden change, determined again by the two parameters a and b, represents a peak of momentary effect. (The user of the program can define the maximum length of time over which a sudden change is still meaningful -- and thus distinguish it from two relatively close step functions of equal size but of opposite directions. At present, such length of time can be one or two time units.)

Finally, the delay function spans over a time period during which the data points cannot be fitted by any morph at the desired level of statistical significance -- the event remains "unexplained" there.

2. AN INFORMAL DESCRIPTION OF THE ALGORITHM

We first give an informal and verbal description of the algorithm developed. A more formal version, written in a

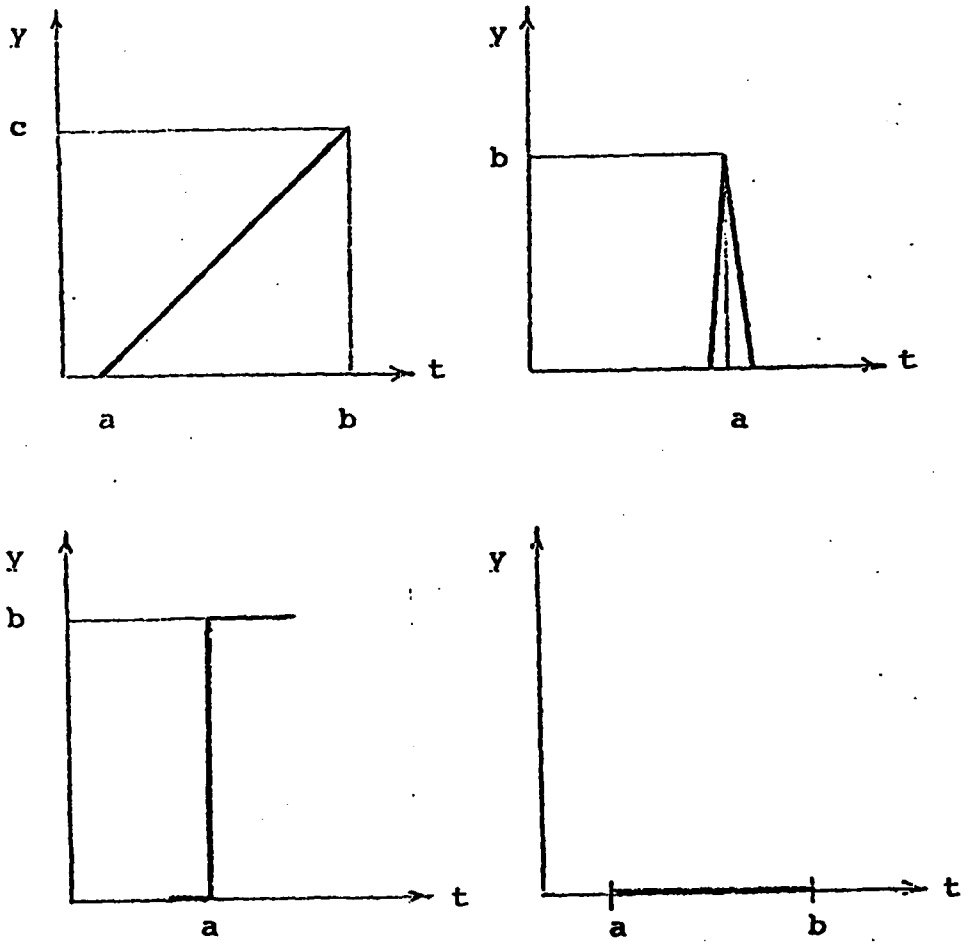


FIGURE 1

The set of parametrized basic patterns or morphs: (1) trend, (2) sudden change, (3) step function and (4) delay function.

PASCAL-like communication language, is in Appendix I.

Let us first define the concept of a window. It has a starting (leftmost) point on the time scale, LOW, and a finishing (rightmost) point, HIGH. The number of datapoints covered in between the two is SCOPE. (Note that two out of these three parameters would suffice to define the window but the algorithm refers to them interchangeably and all are, therefore, continually recomputed. Furthermore, SCOPE rather than the time-difference between HIGH and LOW is meaningful because the datapoints along the time-scale are not necessarily equidistant, as mentioned before.)

Next, we discuss the two statistical measures for the goodness of fit of a trend, the F-ratio and the correlation coefficient R.

$$F = \frac{s^2_{yreg-\bar{y}}}{s^2_{yreg-y}} (N - 2) \qquad (1)$$

where the numerator is the variance of the regression line's Y-values, yreg, around the mean of the Y-values of the datapoints, $\bar{y}$; the denominator is the variance of the Y-values of the datapoints, y, around the regression line's Y-values, yreg. The degrees of freedom is one for the numerator and N-2 for the denominator, with N being the number of datapoints considered for the trend.

The other, alternative measure is

$$R = \pm \frac{\sum_{i=1}^{N} t_i.y_i - N.\bar{t}.\bar{y}}{[(\sum_{i=1}^{N} t_i^2 - N\bar{t}^2)(\sum_{i=1}^{N} y_i^2 - N\bar{y}^2)]^{\frac{1}{2}}} \qquad (2)$$

where $\bar{t}$ and $\bar{y}$ are the mean values of the datapoint coordinates. The sign, by convention, is the same as that of the slope of the regression line. The degree of freedom is N-2. The square of the above, called the 'coefficient of determination', can be put in a form similar to (1),

$$R^2 = 1 - \frac{s^2_{yreg-y}}{s^2_{y-\bar{y}}} \qquad (3)$$

Our program uses tabulated values of F and R (at 5% level of significance) in making decisions as to whether the primary morph, a trend fitted, is acceptable or not.

The total range of the datapoints is being scanned by moving the window along the t-axis in both directions, changing the scope of the window when necessary and performing various computations on the datapoints within the window. We have experimented with several plausible algorithms as to when to change the direction of movement and the scope of the window, and what the computations should be within the window after some difficulties have arisen with certain types of test data. It is best to describe the decision-making machinery of the program in terms of initializing the trend, enlengthening the trend (with sudden changes incorporated in both cases), computing the step function, and computing the delay function. It is also informative to outline our process of discovering the most effective algorithm for the first of these.

(a) The Initialization of a Trend

The user must specify the minimum number of datapoints a trend must fit, MINSCOP. The program tries to establish a new

trend, starting with the first datapoint or with the last datapoint of a previous trend. It is inutiuitively obvious that, for a certain goodness of fit, the more datapoints there are, the higher total variance around the trend is allowed.

In out first approach, the program sets up a window with MINSCOP and slides it along the t-axis until a trend is found. If none is found, the scope of the window is increased and the scanning is repeated. The drawback of this technique is that a longer trend, but with a relatively larger scatter of points, is ignored (not discovered) if a shorter trend with relatively better fitting points is established first.

To avoid this drawback we have tried a second approach in which the program enlarges the window gradually -- while anchoring its leftmost point -- up to a limit as long as a trend cannot be established. If still unsuccessful, the window is reduced to MINSCOP again and shifted by one datapoint to the right where the same process is repeated. It turns out that this approach tends to force the inclusion of the datapoints on the left -- they are disregarded only when even the largest window cannot "absorb" the variance caused by them.

Further difficulties were met in our third approach when we added the concept of sudden changes to the algorithm. The routine identifying them returns the scope of the sudden change (1 or 2 datapoints), deletes them from the computation of the trend whose scope is, however, kept constant by adding on the right new datapoints equal in number to that of the deleted ones. This method did not initialize (correct) trends when the variance

was relatively large and also identified wrong sudden changes. This can be easily seen from Figure 2. Suppose MINSCOP=5. No trend is acceptable through the points 1, 2, 3, 4 and 5 (the variance is too large). The program drops point 5 -- assuming it is a sudden change -- and fits a trend through points 1, 2, 3, 4, and 6. However, when all the points from 1 to 8 are considered together through a window of larger scope, point 5 does form a part of the trend.

- -

FIGURE 2 ABOUT HERE

- -

Our final, fourth approach conbimes the first two while avoiding the drawbacks of either. Also, it postpones the identification of sudden changes until after local scanning. It makes sure that the number of datapoints that can be excluded on the left is at most one less than (the current) MINSCOP (i.e. always less than required for a separate trend). However, in order to take into consideration the possibility of "bad" points there, the program will gradually increase MINSCOP up to the total range as long as no trend is found. The process of changing the size of and shifting the window is as follows.

Suppose there is a sequence of datapoints left:

L, L+1, L+2, ... N

Let the initial minimum scope of a trend be

S=MINSCOP

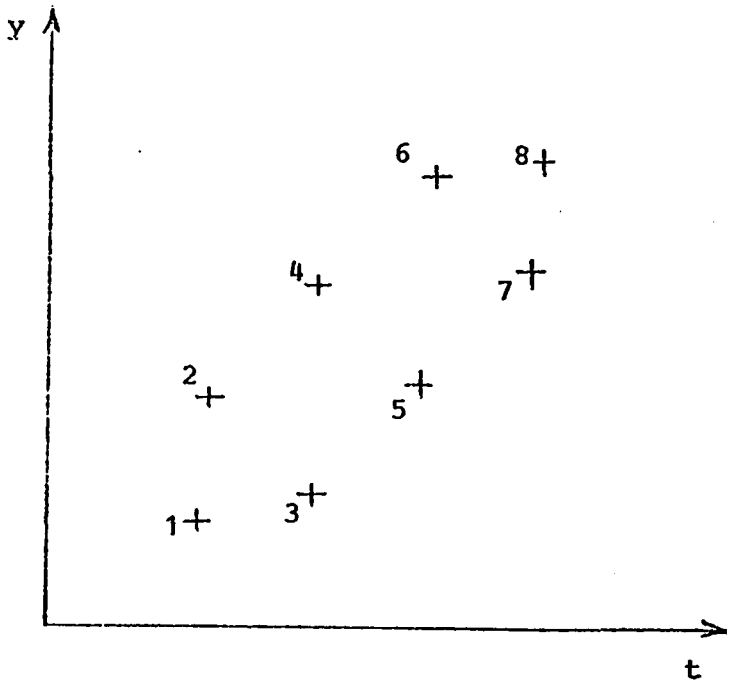


FIGURE 2
A problem in trend initialization (see text).

The local scanning always has the range between L and L+2S-2 ($\leq$N) so that when the window is in the rightmost position, there can be only S-1 datapoints on the left that do not belong to a trend potentially established across the window. The span of the window is shifted step-by-step (S-1) times through

(L, L+S -1)

(L+1, L+S)

(L+2, L+S+1)

.

.

.

(L+S+1, L+2S-2)

If no trend is found, the window span, S, is increased and the process of "local scanning" is repeated. This goes on until either a trend is found or the window span becomes so large that the last datapoint is its right boundary.

As can be seen, this algorithm uses the first approach iteratively, within the range of local scanning. There is a certain amount of redundant search, when the same window span is used over the same datapoints at different local scans, but the process is still very efficient and the drawbacks of the first two approaches are avoided.

The trend so initialized is only the starting point. The system tries to make it optimally long, which is the task of the next phase.

(b) <u>The Lengthening of a Trend</u>

Once a trend has been initialized, the system tries to

enlengthen its range as much as possible until a state of satisfactory fit is reached. First, it adds points on the right as long as the quality of fit does not deteriorated below a threshold value. Then it tries to drop one or two "bad" points on the left if doing so improves the fit above (another) threshold value. If dropping of points on the left has taken place, the whole process starts anew: adding points at the right and dropping points at the left in a ping-pong fashion until an equilibrium is arrived at. Then the direction of iteration is reversed and points are added at the left if possible.

Care has to be taken, however, with adding on the left points that currently (and tentatively) belong to the previous trend. The decision criterion is that the point in question is allocated to that trend whose quality of fit improves more by including the point. If the point represented a sudden change with the previous trend and it would improve the fit of the current trend, then it is simply annexed by the latter.

In the process of re-allocating points between trends, the parameters are of course recomputed. If the scope of the shortened trend becomes smaller than the minimum allowed, the trend is deleted or "swallowed up" by the subsequent one. As can be seen, the system is extremely dynamic and can go through a fairly large number of iterations from left to right and back until an optimum arrangement is obtained. Using judiciously selected levels for the threshold values noted above, the convergence process is very fast.

(c) Computing the Step Function

There are two distinct cases of identifying the size of the step function. When there is a delay function (see below) between two subsequent trends, the value sought is the difference between the y-values of the last point of the previous trend and of the first point of the current trend.

When there is no delay function, the trends are adjacent. These can either quasi share a point (being within the scope of both), as shown on Figure 3a, or they are vertically separated at the boundary of their respective scopes by the step size, as shown on Figure 3b. (By the way, different algorithms have to take care of the two cases of zero delay described below.)

- -

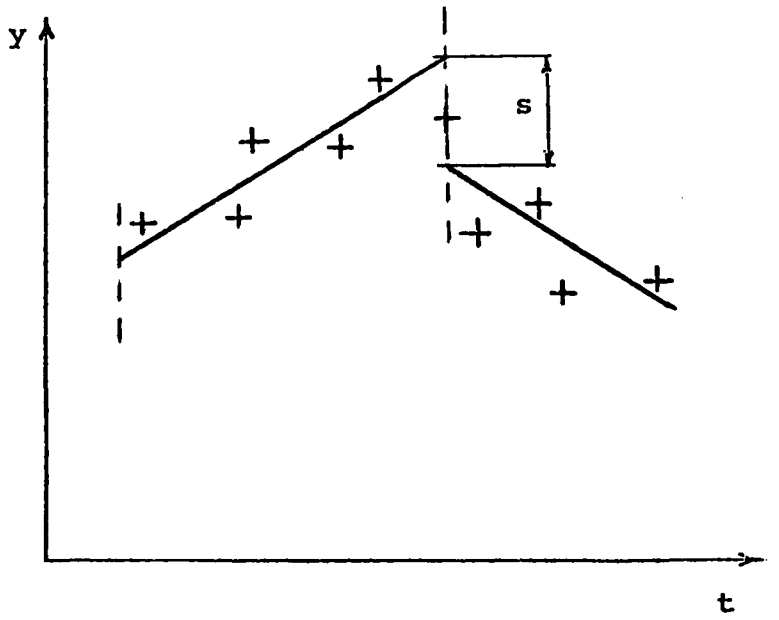
FIGURES 3a AND 3b ABOUT HERE

- -

(d) Computing the Delay Function

As discussed earlier, the delay function covers a time period over which no mathematical interpretation of the events is possible (with the accepted level of statistical significance). The "value" of the delay function equals the number of datapoints within the period.

There are two possible cases of zero delay. The scopes of two adjacent trends meet at the time coordinate of a datapoint (which is quasi shared by them) or in between two datapoints which thus belong to different trends. These are shown on Figures 4a and b, respectively.



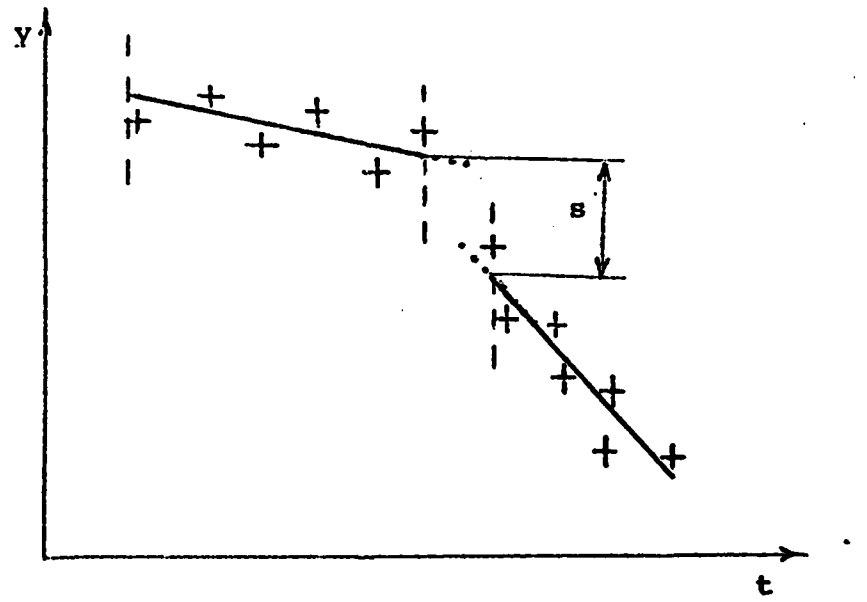


FIGURE 3

Two cases of computing the size of the step function, s, with zero derlays between adjacent trends.

- -

FIGURES 4a AND 4b ABOUT HERE

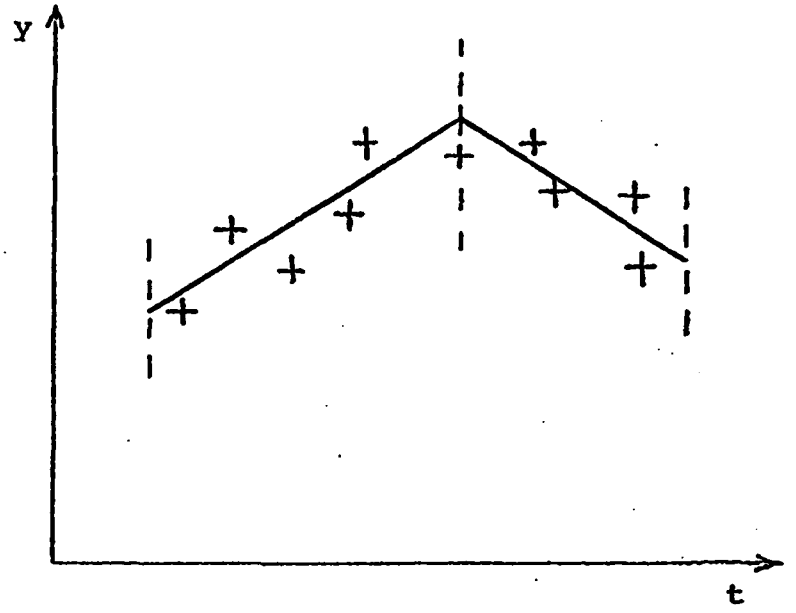
- -

3. THE PROGRAM

The program, written in CDC Extended FORTRAN, comprises about 1,100 lines. The user specifies, in addition to the coordinates of the datapoints, the minimum scope of a trend, and the threshold values for adding and dropping points at the two ends of a trend (high and low end values of the correlation coefficient R and the F-ratio.)

In trace mode, the program provides an account of finding each morph by printing the tentative parameters of every iteration, and outputs messages and statistical measures of the quality of fit, and the final values of all above. (The selective trace prints out only the essential and final information.) If the plotting mode is selected, both the datapoints and the morphs are outputted on a plotter (in addition to the lineprinter, which is always used for the datapoints.. The results of statistical calculations and comparisons concerning the goodness of fit are printed together with the final specification of each morph fitted; regardless of the trace mode used

To give a rough estimate of the time required for computations, the following CYBER 173 CPU times were spent on the tasks represented by Figures 5-10 (in addition to 9.891s compilation time):



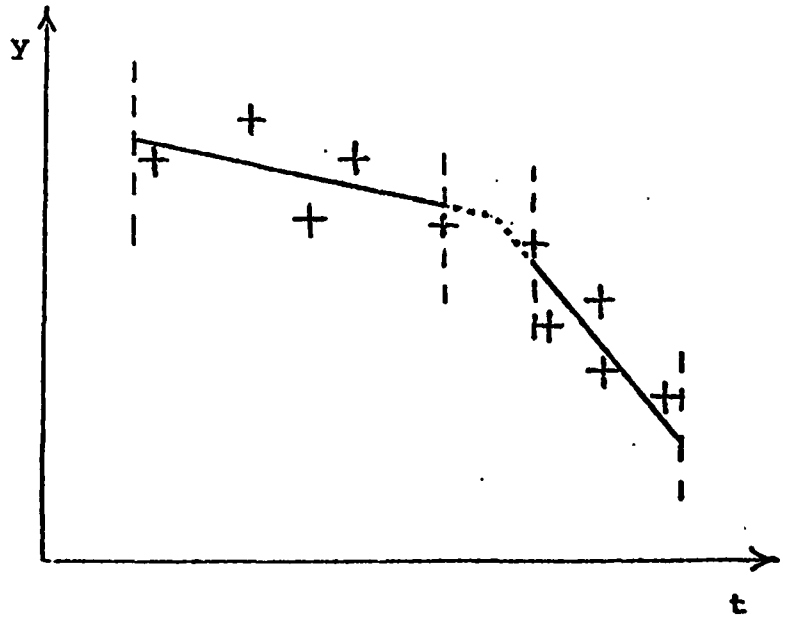


FIGURE 4
Two cases of zero delays between adjacent trends.

Task No.	Figure No.	CPU time in ms
1	5	987
2	6	860
3	7	872
4	8	759
5	9	836
6	10	977

FIGURES 5 TO 10 ABOUT HERE

4. SUMMARY

We have developed a flexible and effective technique to describe a time-sequence of points mathematically, in terms of an indefinite number of basic patterns or morphs. The technique has been shown to be more general than multiphase (segmented) linear regression analysis. The amount of noise tolerated around the morphs can easily be adjusted by changing certain sets of tabulated values and user-provided parameters.

5. ACKNOWLEDGEMENT

We acknowledge the support of AFOSR grant 81-0220.

6. REFERENCES

[1] Quandt, R.E.: Tests of the hypothesis that a linear regression system obeys two separate regimes. (J. Am. Statist. Assoc., 55, pp. 324-330, 1960).

[2] Hinkley, D.V.: Inference about the intersection in two-phase regression (Biometrika, 56, pp. 495-504, 1969).

[3] Bellman, R. and R. Roth: Curve fitting by segmented

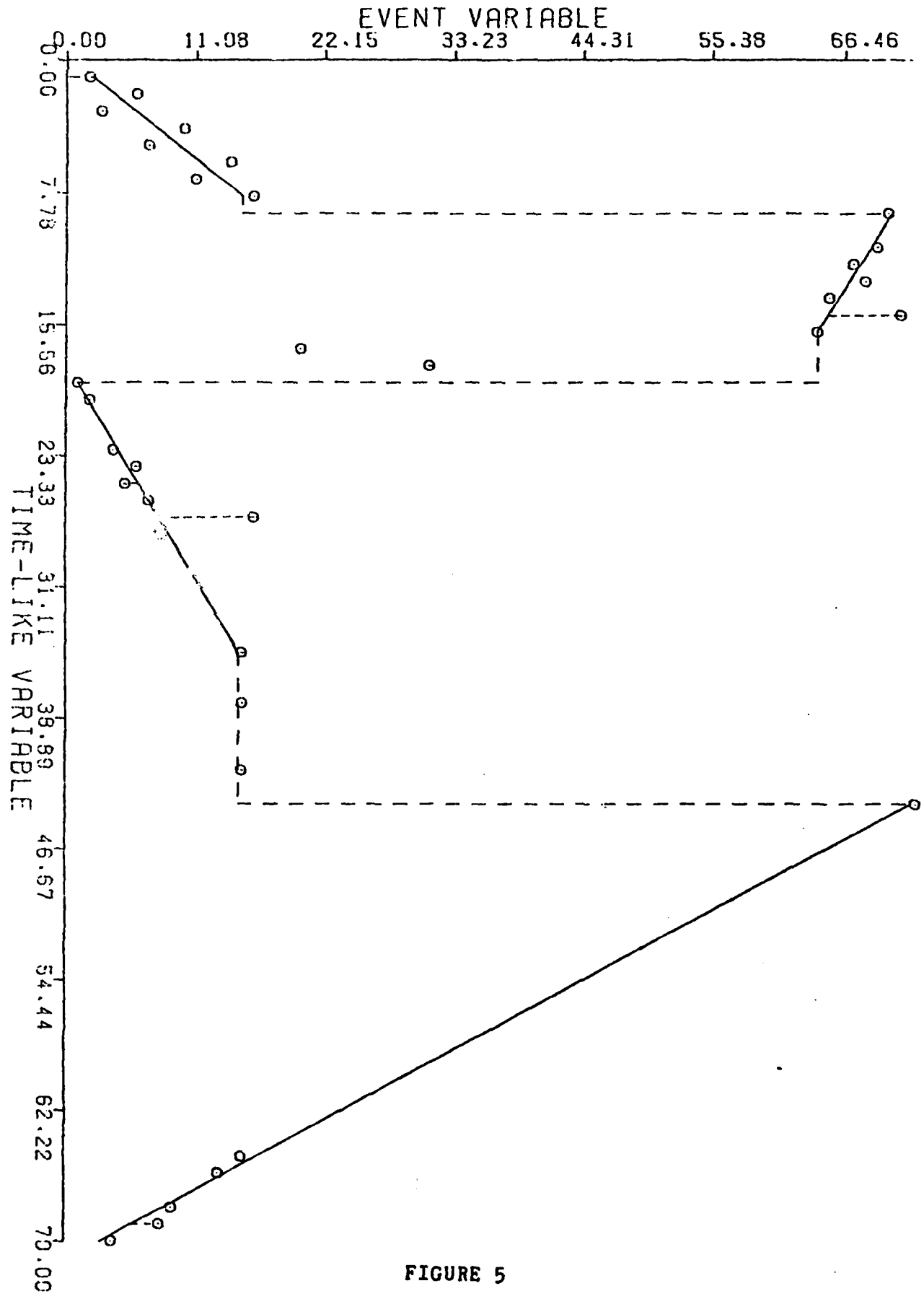

EVENT VARIABLE
0.00
11.08
22.15
33.23
44.31
55.38
66.46
TIME-LIKE VARIABLE
0.00
7.78
15.56
23.33
31.11
38.89
46.67
54.44
62.22
70.00

FIGURE 5

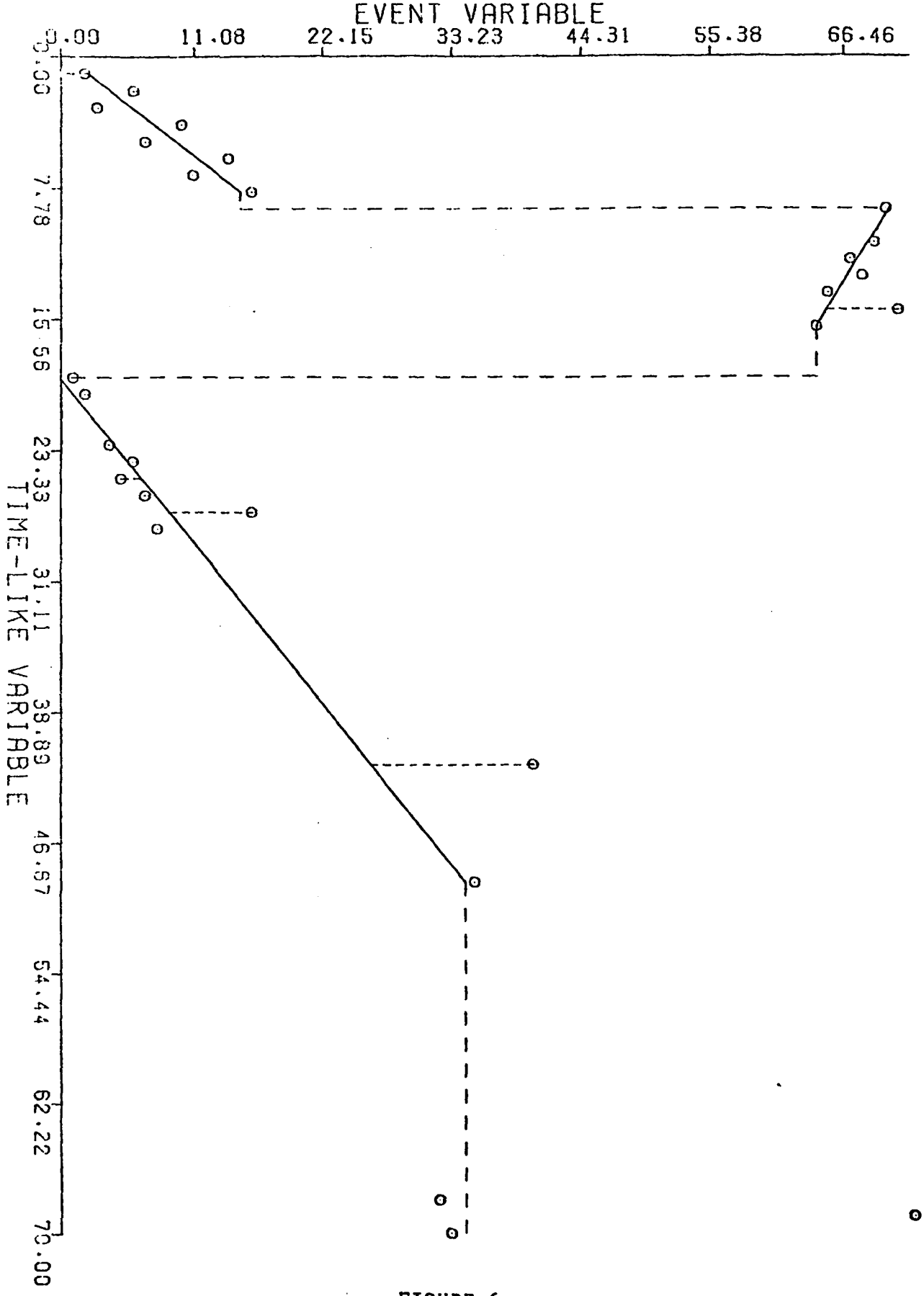

EVENT VARIABLE
0.00
11.08
22.15
33.23
44.31
55.38
66.46
TIME-LIKE VARIABLE
0.00
7.78
15.56
23.33
31.11
38.89
46.67
54.44
62.22
70.00

FIGURE 6

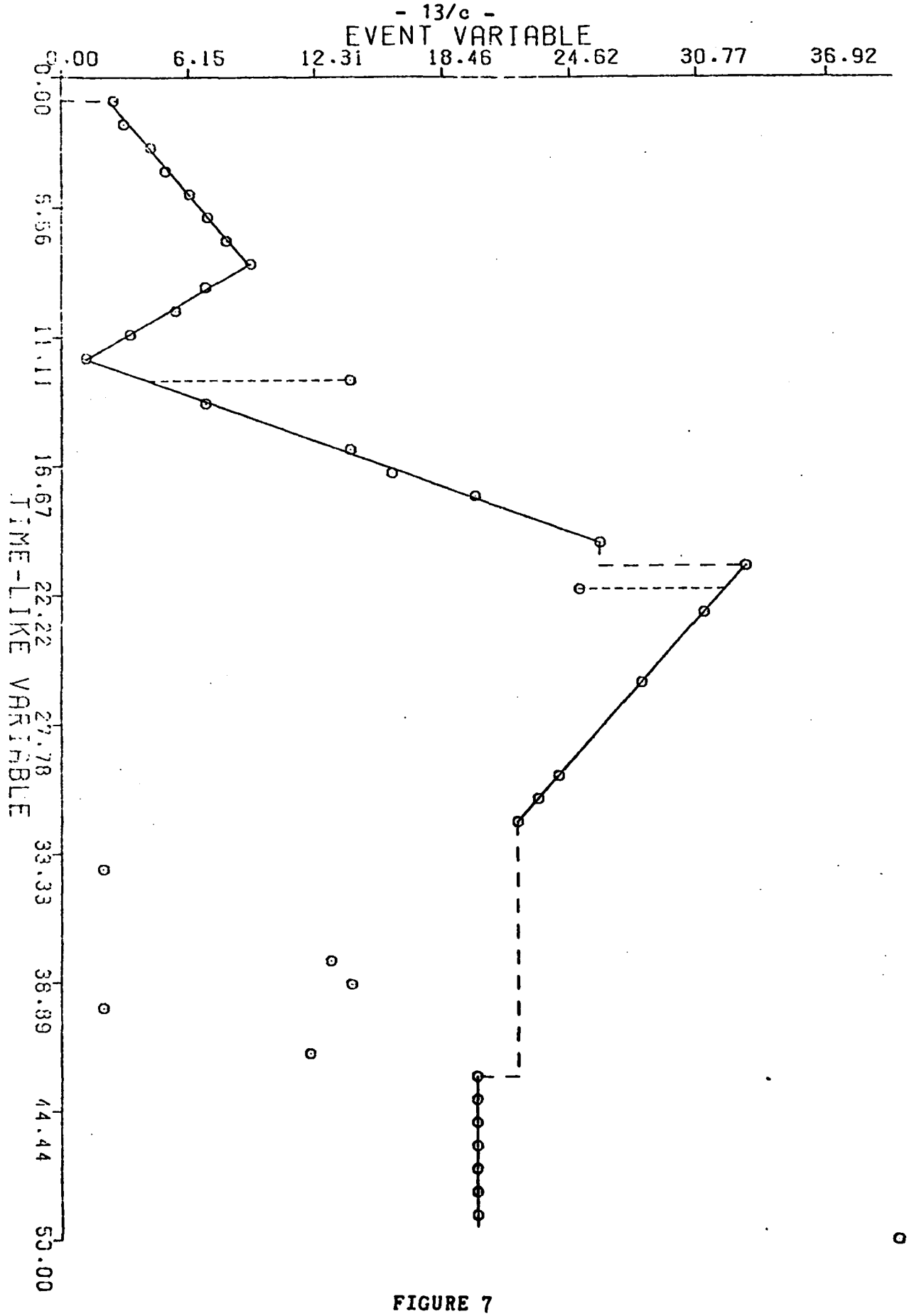

EVENT VARIABLE
0.00
6.15
12.31
18.46
24.62
30.77
36.92
TIME-LIKE VARIABLE
0.00
5.56
11.11
16.67
22.22
27.78
33.33
38.89
44.44
50.00

FIGURE 7

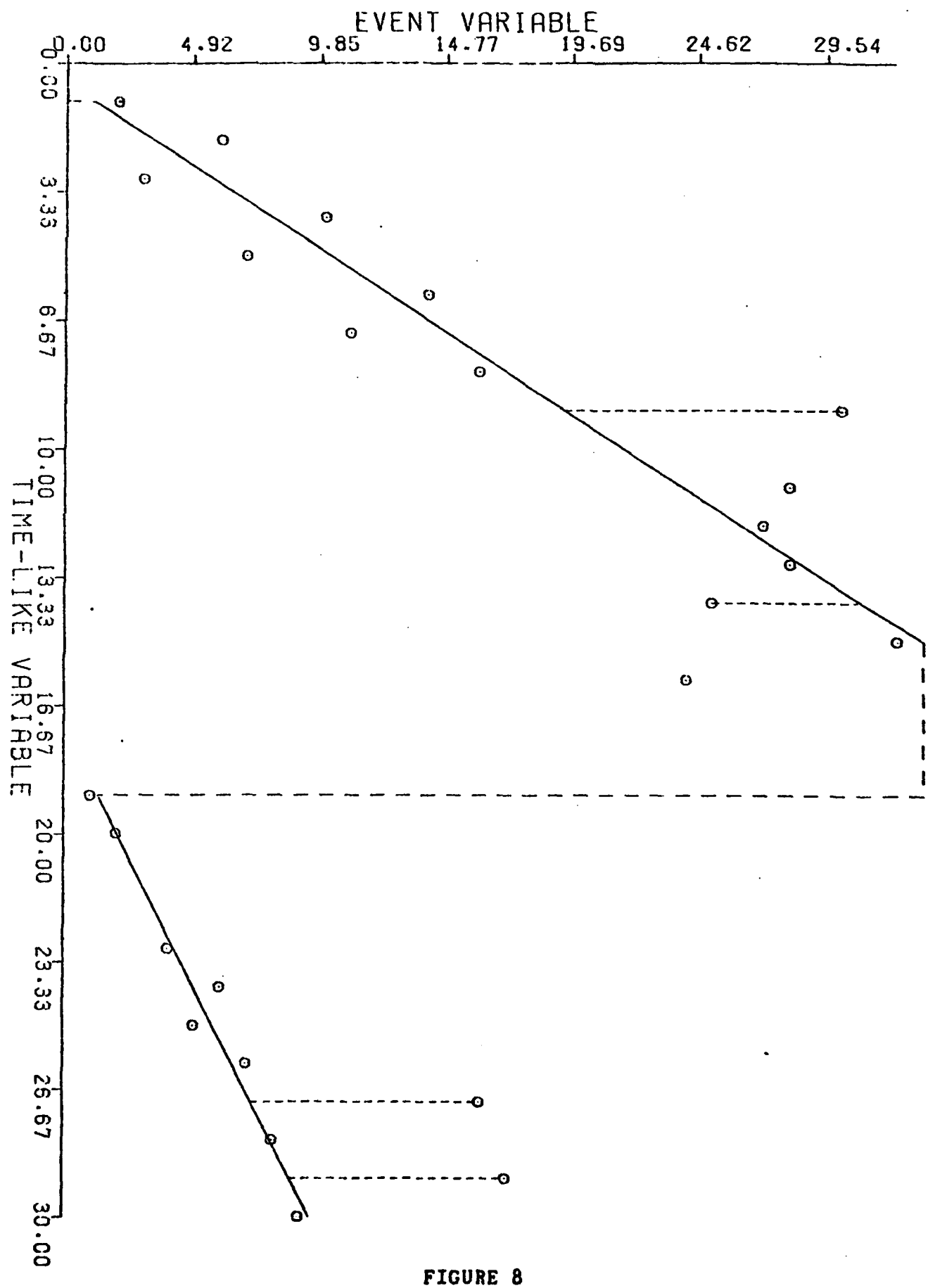

EVENT VARIABLE
0.00
4.92
9.85
14.77
19.69
24.62
29.54
TIME-LIKE VARIABLE
0.00
3.33
6.67
10.00
13.33
16.67
20.00
23.33
26.67
30.00

FIGURE 8

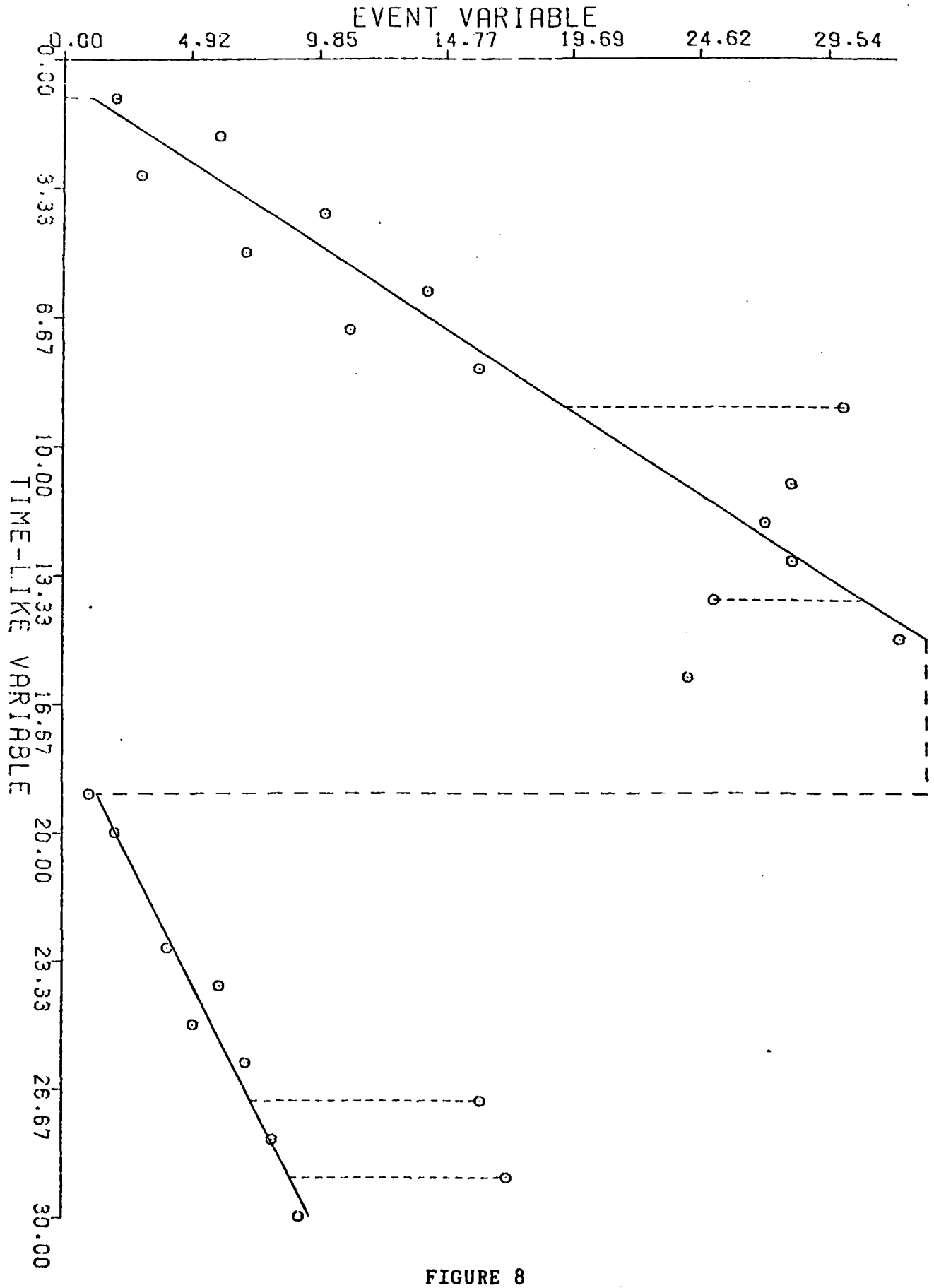

EVENT VARIABLE
0.00
4.92
9.85
14.77
19.69
24.62
29.54
TIME-LIKE VARIABLE
0.00
3.33
6.67
10.00
13.33
16.67
20.00
23.33
26.67
30.00

FIGURE 8

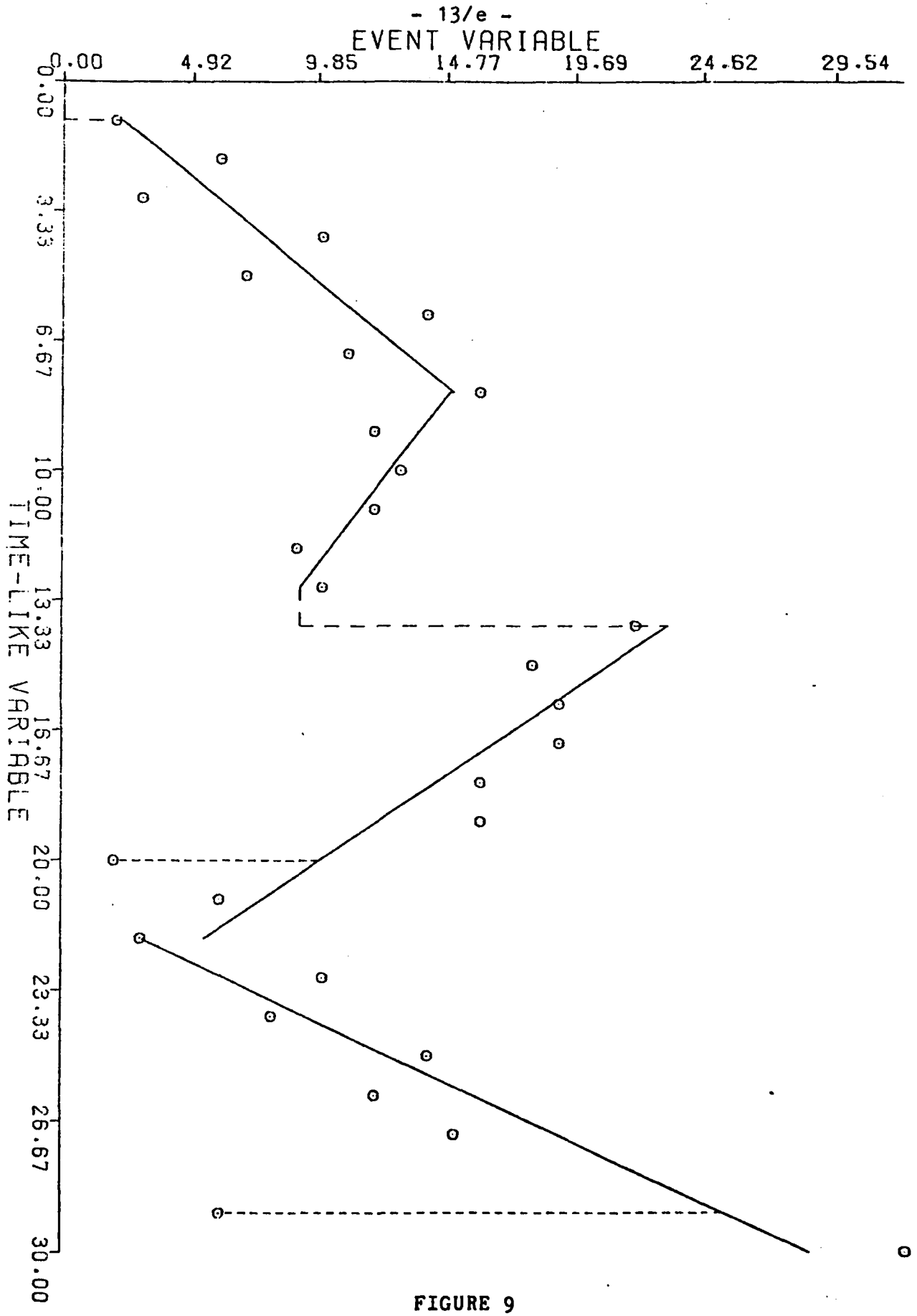

EVENT VARIABLE
0.00
4.92
9.85
14.77
19.69
24.62
29.54
TIME-LIKE VARIABLE
0.00
3.33
6.67
10.00
13.33
16.67
20.00
23.33
26.67
30.00

FIGURE 9

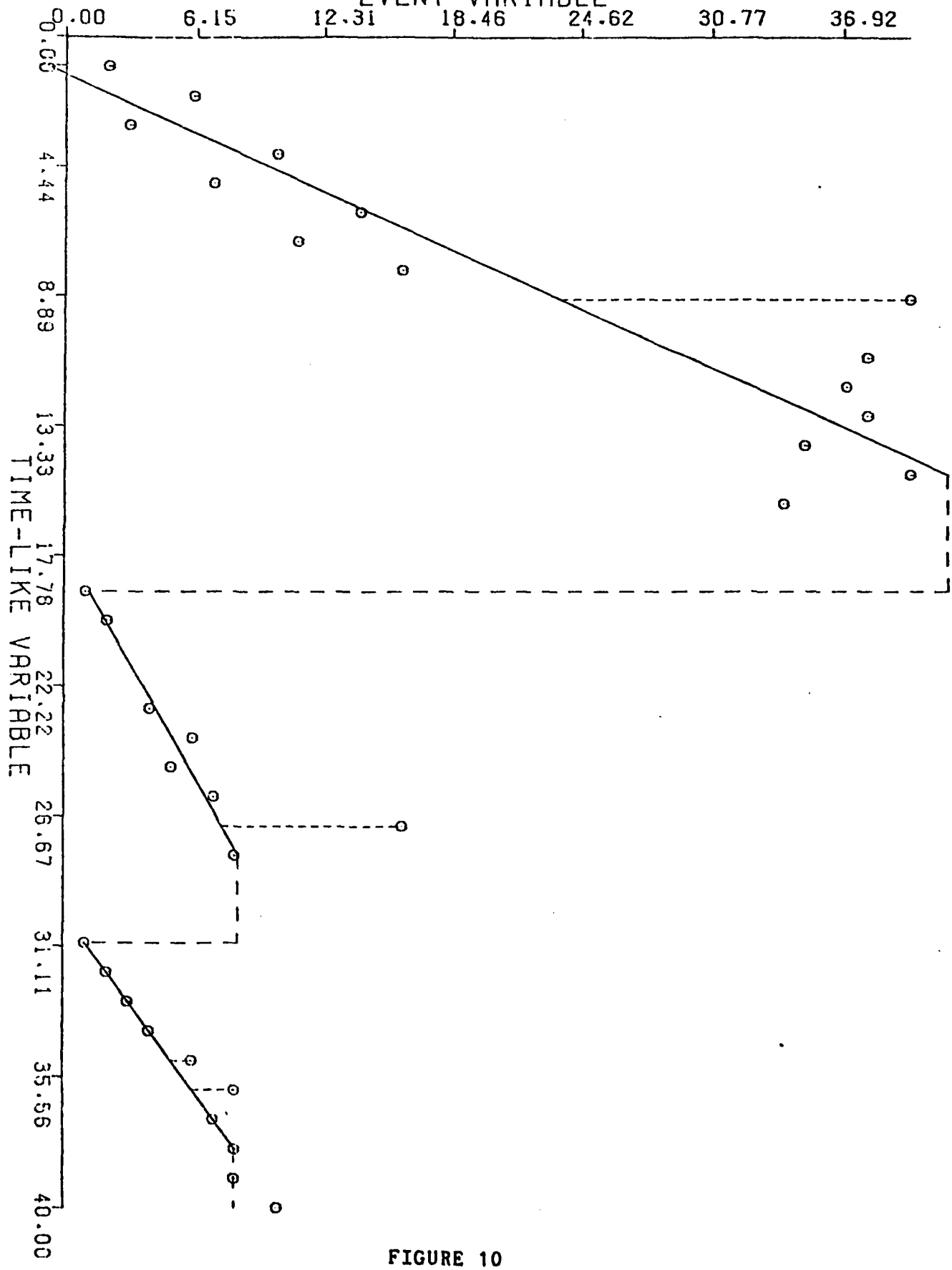

EVENT VARIABLE
0.00
6.15
12.31
18.46
24.62
30.77
36.92
TIME-LIKE VARIABLE
0.00
4.44
8.89
13.33
17.78
22.22
26.67
31.11
35.56
40.00

FIGURE 10

straight lines (J. Am. Statist. Assoc., 64, pp. 1079-1084, 1964).

[4] Feder, P.I.: The log likelihood ratio on segmented regression (Ann. Statist., 3, pp. 84-97, 1975).

[5] Beckman, R.J. and R.D. Cook: Testing for two-phase regression (Technometrics, 21, pp. 65-69, 1979).

[6] Findler, N.F., H. Klein, W. Gould, A. Kowal, and J. Menig: Studies on decision-making using the game of Poker (Proc. IFIP Congress 71, Ljubljana, Yugoslavia, pp. II/191-II/194, 1970).

[7] Findler, N.F., H. Klein, and Z. Levine: Experiments with inductive discovery processes leading to heuristics in a Poker program (In Beckmann, Goos and Künzi (Eds.): Cognitive Processes and Systems. Springer: Berlin, 1973).

[8] Findler, N.V.: Computer experiments on forming and optimizing heuristic rules (In Elithorn and Jones (Eds.): Artificial and Human Thinking. Elsevier: Amsterdam, 1973).

[9] Findler, N.V., H. Klein, R.C. Johnson, A. Kowal, Z. Levine, and J. Menig: Heuristic programmers and their gambling machines (Proc. ACM National Conf., San Diego, CA, 1974, pp. 28-37).

[10] Findler, N.V.: Studies in machine cognition using the game of Poker (Comm. ACM, 20, pp. 230-245, 1977).

[11] Findler, N.V.: Computer Poker (Scientific American, pp. 144-151, 1978).

[12] Findler, N.V. and J. van Leeuwen: The complexity of decision trees, the Quasi-Optimizer, and the power of the heuristic rules (Information and Control, 40, pp. 1-19, 1979).

[13] Findler, N.V., G. Sicherman, and S. Feuerstein: Teaching strategies to an Advice Taker/Inquirer system (Proc. Euro IFIP 79

Conf., London, England, pp. 457-465, 1979).

[14] Findler, N.V.: Aspects of computer learning (Cybernetics and Systems, 11, pp. 65-84, 1980).

[15] Findler, N.V.: Pattern recognition and generalized production systems in strategy development (Proc. Fifth Internat. Conf. on Pattern Rec., Vol. I, pp. 140-145, 1980).

[16] Findler, N.V.: A multi-level learning technique using production systems (Submitted for publication).

APPENDIX

The following is a very concise, high-level definition of the main algorithms used in a PASCAL-like communication language:

```
program fit;
begin
    while not end-of-sequence do
      if possible-to-init-a-trend then
        begin
           enlarge-it;
           compute-the-delay;
           compute-the-step;
        end;
end.
function possible-to-init-a-trend:  logical;
begin
    set-initial-minimum-scope-window
    while possible and trend-not-found do
      begin
        while ((not local-scanning) and (sudden-changed=0) and
              possible) do
           increase-scope-of-window-to-the-right;
        if trend-not-found
           then
             begin
                restore-the-minimum-scope-window
                slide-it-to-the-right;
             end;
      end;
end;
```

```
function local-scanning:logical;
begin
    repeat
       set-window-at-left-extreme-of-interval;
       while (possible and (not init-test)) do
           slide-window-to-the-right;
       if trend-not-found
           then increase-scope-of-window;
    until (not possible) or trend-found;
end;

procedure enlarge;
begin
    repeat
       add-points-on-right;
    until not drop-points-on-left(2) or drop-points-on-left(1);
    add-points-to-the-left;
end;

procedure add-points-on-right;
begin
    repeat;
       while possible and fit-does-not-deteriorate do
           increase-scope-of-window;
    until sudden-changes=0;
end;
```

```
procedure add-points-on-left;
begin
    flag:=true;
    repeat
       while possible and (not conflicting-with-previous-trend)
           and fit-does-not-deteriorate do
           increase-scope-of-window-to-left;
       if conflicting-with-previous-trend
         then
           if conflicting-point-is-a-sudden-change-of-last-trend
             then
                if fit-does-not-deteriorate
                    then delete-point-from-previous-trend
                    else flag:=false
             else
                begin
                  test:=improvement-of-quality-
                        of-fit-in-current-trend;
                  test1:=improvement-of-quality-
                         of-fit-in-previous-trend;
                    if test>test1
                      then delete-point-from-previous-trend
                      else flag:=false;
    until not flag;
end;

function drop-points-on-left(n:integer):logical;
```

```
begin
    if scope<=(minimumscope+n-1)
       then drop-points-on-left:=false
       else
         begin
           low:=low-n;
           scope:=scope-n;
           if fit-does-improve
             then drop-points-on-left:=true
             else
                begin
                     drop-points-on-left:=false;
                     low:=low-n;
                     scope:=scope+n;
                end;
end;
```

```
procedure delay;
begin
    if first-trend
       then delay:=number-of-points-left-unexplained-
                   before-this-trend;
       else delay:=number-of-points-left-unexplained-
                   between-last-trend-and-this-trend;
end;
procedure step;
begin
    if first-trend
       then step:=regression-value-of-first-point-in-trend
       else
           begin
           yleft:=regression-value-of(left);
                  {left-last-point-on-last-trend}
           yright:=regression-value-of (right);
                   {right-first-point-on-this-trend}
           if delay≠0
              then step:=yright-yleft
              else if left-is-accepted-by-this-trend and right-is-
                      accepted-by-last-trend
                    then step:=0
                    else step:=yright-yleft;
           end;
end;
```

LEGEND FOR FIGURES:

FIGURE 1 - The set of parametrized basic patterns or morphs: (1) trend, (2) sudden change, (3) step function and (4) delay function.

FIGURE 2 - A problem in trend initialization (see text).

FIGURE 3 - Two cases of computing the size of the step function, s, with zero delays between adjacent trends

FIGURE 4 - Two cases of zero delays between adjacent trends

FIGURES 5 TO 10 - Six exemplary tasks for the morph-fitting program.

FOOTNOTE (ON PAGE 1)

The development of the algorithm was joint effort. The program based on it was produced by E.M. and the paper was written by N.V.F.

INDEX TERMS:

Approximation techniques,

Rules of pattern formation,

Simplified time-series analysis,

Multiphase (sequential) regression analysis,

Controlled statistical measures for approximation.